PATENT ABSTRACTS OF JAPAN

(11)Publication number : 11-007387

(43)Date of publication of application : 12.01.1999

(51)Int.Cl. G06F 9/38
G06F 9/30

(21)Application number : 09-159048

(22)Date of filing : 16.06.1997

(71)Applicant : MATSUSHITA ELECTRIC IND CO LTD

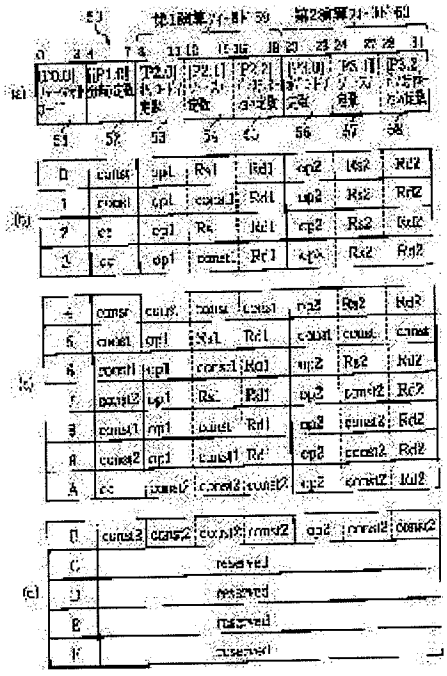
(72)Inventor : TAKAYAMA SHUICHI
HIGAKI NOBUO

(54) VLIW PROCESSOR

(57)Abstract:

PROBLEM TO BE SOLVED: To provide a VLIW(very long instruction word) processor to execute an instruction structured with excellent code efficiency with comparatively short word and capable of simultaneously specifying many operations.

SOLUTION: Fields 52, 59, 60 to specify three pieces of operation at maximum to the instruction 50 of 32 bit length is provided. Only an operation code 'cc' to specify branch operation in which a stored value of an implicitly specified constant register is defined as a branching destination address or a constant 'const' to be set in the constant register is placed in a P 1.0 field 52. Which of them is placed is specified by a formal code placed in a P 0.0 field 51 of four bit length.



(19)日本国特許庁（ＪＰ）　(12)**公開特許公報**（Ａ）　(11)特許出願公開番号

特開平11－7387

(43)公開日　平成11年(1999) 1月12日

(51)Int.Cl.[6]		識別記号	FI		
Ｇ０６Ｆ	9/38	３１０	Ｃ０６Ｆ	9/38	３１０Ａ
					３１０Ｈ
	9/30	３５０		9/30	３５０Ｆ

審査請求　未請求　請求項の数27　ＯＬ　（全 18 頁）

(21)出願番号　特願平9－159048

(22)出願日　平成９年(1997) ６月16日

(71)出願人　000005821
松下電器産業株式会社
大阪府門真市大字門真1006番地

(72)発明者　高山　秀一
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(72)発明者　槍垣　信生
大阪府門真市大字門真1006番地　松下電器産業株式会社内

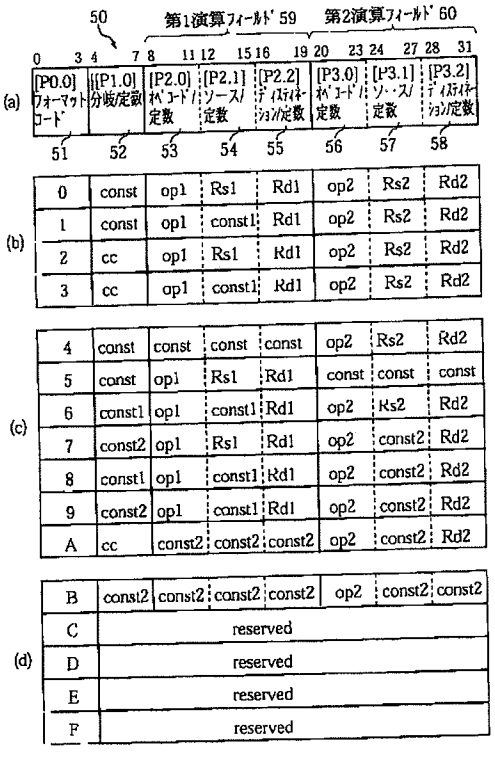
(74)代理人　弁理士　中島　司朗

(54)【発明の名称】　ＶＬＩＷプロセッサ

(57)【要約】

【課題】　比較的短い語長の命令であって、かつ、同時に多くのオペレーションを指定することができるコード効率のよい構造を有する命令を実行するＶＬＩＷプロセッサを提供する。

【解決手段】　３２ビット長の命令５０に最大３個のオペレーションを指定するフィールド５２、５９、６０を設ける。Ｐ１．０フィールド５２には、暗黙的に指定された定数レジスタ３６の格納値を分岐先番地とする分岐オペレーションを指定するオペコード”ｃｃ”のみ、又は、定数レジスタ３６にセットする定数“ｃｏｎｓｔ”が置かれる。いずれが置かれているかは、４ビット長のＰ０．０フィールド５１に置かれたフォーマットコードによって特定される。



【特許請求の範囲】
【請求項１】　複数のオペレーションフィールドからなる命令を実行するＶＬＩＷプロセッサにおいて、
前記オペレーションフィールドの大きさが不均一であり、かつ前記命令の命令語長は前記命令の持つオペレーションフィールドの数で割り切れないものであることを特徴とするＶＬＩＷプロセッサ。
【請求項２】　複数のオペレーションフィールドからなる命令を実行するＶＬＩＷプロセッサにおいて、
前記オペレーションフィールドの大きさが不均一であり、かつ前記命令は３つのオペレーションフィールドを３２ビットの命令語長中に持つものであることを特徴とするＶＬＩＷプロセッサ。
【請求項３】　複数のオペレーションフィールドからなる命令を実行するＶＬＩＷプロセッサにおいて、
前記オペレーションフィールドのうち少なくとも１つはオペランドの数が異なるものであり、かつ前記命令の命令語長は前記命令の持つオペレーションフィールドの数で割り切れないものであることを特徴とするＶＬＩＷプロセッサ。
【請求項４】　複数のオペレーションフィールドからなる命令を実行するＶＬＩＷプロセッサにおいて、
前記オペレーションフィールドのうち少なくとも１つはオペランドの数が異なるものであり、かつ前記命令は３つのオペレーションフィールドを３２ビットの命令語長中に持つものであることを特徴とするＶＬＩＷプロセッサ。
【請求項５】　複数のオペレーションフィールドからなる命令を実行するＶＬＩＷプロセッサにおいて、
前記オペレーションフィールドのうち１つはオペコードのみからなり、かつ前記命令の命令語長は前記命令の持つオペレーションフィールドの数で割り切れないものであることを特徴とするＶＬＩＷプロセッサ。
【請求項６】　複数のオペレーションフィールドからなる命令を実行するＶＬＩＷプロセッサにおいて、
前記オペレーションフィールドのうち１つはオペコードのみからなり、かつ前記命令は３つのオペレーションフィールドを３２ビットの命令語長中に持つものであることを特徴とするＶＬＩＷプロセッサ。
【請求項７】　２個以上のオペレーションフィールドを含む命令を解読し実行するＶＬＩＷプロセッサであって、
第１の前記オペレーションフィールドにはオペレーションの種類を指定する１個のオペコードのみが置かれ、
第２の前記オペレーションフィールドには１個のオペコードとオペレーションの対象となるデータを指定する１個以上のオペランドとの組が置かれ、
前記第１のオペレーションフィールドに置かれたオペコードを解読する第１の解読手段と、
前記第１の解読手段による解読結果に基づいて前記オペコードによって指定されたオペレーションを実行する第１の実行手段と、
前記第２のオペレーションフィールドに置かれたオペコードを解読する第２の解読手段と、
前記第２の解読手段による解読結果に基づいて前記オペランドによって指定されたデータに対して前記オペコードによって指定されたオペレーションを実行する第２の実行手段とを備えることを特徴とするＶＬＩＷプロセッサ。
【請求項８】　前記第１のオペレーションフィールドの桁数は前記第２のオペレーションフィールドの桁数よりも小さいことを特徴とする請求項７記載のＶＬＩＷプロセッサ。
【請求項９】　前記第１のオペレーションフィールドに置かれたオペコードの桁数は前記第２のオペレーションフィールドに置かれたオペコードの桁数と等しいことを特徴とする請求項８記載のＶＬＩＷプロセッサ。
【請求項１０】　前記命令に含まれるオペレーションフィールドは３個であり、
第３の前記オペレーションフィールドは前記第２のオペレーションフィールドと同じ桁数であって１個のオペコードと１個以上のオペランドとの組が置かれ、
前記ＶＬＩＷプロセッサはさらに、
前記第３のオペレーションフィールドにオペコードが置かれた場合に前記オペコードを解読する第３の解読手段と、
前記第３の解読手段による解読結果に基づいて前記オペランドによって指定されたデータに対して前記オペコードによって指定されたオペレーションを実行する第３の実行手段とを備えることを特徴とする請求項９記載のＶＬＩＷプロセッサ。
【請求項１１】　前記第１の実行手段は、実行すべき命令の流れを制御することを特徴とする請求項１０記載のＶＬＩＷプロセッサ。
【請求項１２】　前記第２の実行手段は、前記第２のオペランドフィールドに置かれたオペランドによって指定されたデータの転送を制御し、
前記第３の実行手段は、前記第３のオペランドフィールドに置かれたオペランドによって指定されたデータの算術論理演算を実行することを特徴とする請求項１１記載のＶＬＩＷプロセッサ。
【請求項１３】　２個以上のオペレーションフィールドを含む命令を解読し実行するＶＬＩＷプロセッサであって、
第１の前記オペレーションフィールドにはオペレーションの種類を指定する１個のオペコードのみ又は定数のみが置かれ、
第２の前記オペレーションフィールドには１個のオペコードとオペレーションの対象となるデータを指定する１個以上のオペランドとの組又は定数のみが置かれ、

前記第１のオペレーションフィールドにオペコードが置かれた場合に前記オペコードを解読する第１の解読手段と、
前記第１の解読手段による解読結果に基づいて前記オペコードによって指定されたオペレーションを実行する第１の実行手段と、
前記第２のオペレーションフィールドにオペコードが置かれた場合に前記オペコードを解読する第２の解読手段と、
前記第２の解読手段による解読結果に基づいて前記オペランドによって指定されたデータに対して前記オペコードによって指定されたオペレーションを実行する第２の実行手段とを備えることを特徴とするＶＬＩＷプロセッサ。
【請求項１４】　前記命令はさらに、前記第１及び第２のオペレーションフィールドそれぞれに定数のみが置かれているか否かを指定するフォーマットコードが置かれたフォーマットフィールドを含み、
前記ＶＬＩＷプロセッサはさらに、
前記フォーマットコードを解読するフォーマット解読手段と、
前記フォーマット解読手段により前記第１、第２及び第３の少なくとも１つのオペレーションフィールドに定数のみが置かれていると解読された場合に、その定数を取り出して記憶する定数記憶手段とを備えることを特徴とする請求項１３記載のＶＬＩＷプロセッサ。
【請求項１５】　前記第１のオペレーションフィールドの桁数は前記第２のオペレーションフィールドの桁数よりも小さいことを特徴とする請求項１４記載のＶＬＩＷプロセッサ。
【請求項１６】　前記第１のオペレーションフィールドに置かれたオペコードの桁数は前記第２のオペレーションフィールドに置かれたオペコードの桁数と等しいことを特徴とする請求項１５記載のＶＬＩＷプロセッサ。
【請求項１７】　前記命令に含まれるオペレーションフィールドは３個であり、
第３の前記オペレーションフィールドは前記第２のオペレーションフィールドと同じ桁数であって１個のオペコードと１個以上のオペランドとの組が置かれ、
前記ＶＬＩＷプロセッサはさらに、
前記第３のオペレーションフィールドにオペコードが置かれた場合に前記オペコードを解読する第３の解読手段と、
前記第３の解読手段による解読結果に基づいて前記オペランドによって指定されたデータに対して前記オペコードによって指定されたオペレーションを実行する第３の実行手段とを備えることを特徴とする請求項１６記載のＶＬＩＷプロセッサ。
【請求項１８】　前記第１の実行手段は、実行すべき命令の流れを制御することを特徴とする請求項１７記載のＶＬＩＷプロセッサ。
【請求項１９】　前記第２の実行手段は、前記第２のオペランドフィールドに置かれたオペランドによって指定されたデータの転送を制御し、
前記第３の実行手段は、前記第３のオペランドフィールドに置かれたオペランドによって指定されたデータの算術論理演算を実行することを特徴とする請求項１８記載のＶＬＩＷプロセッサ。
【請求項２０】　前記フォーマットフィールドの桁数、前記第１のオペレーションフィールドの桁数、前記第２及び第３のオペレーションフィールドに置かれたオペコードの桁数、前記第２及び第３のオペレーションフィールドに置かれた各オペランドの桁数は、いすれもｎビットであることことを特徴とする請求項１９記載のＶＬＩＷプロセッサ。
【請求項２１】　前記命令は３２ビット長であり、
前記ｎは４であることを特徴とする請求項２０記載のＶＬＩＷプロセッサ。
【請求項２２】　３個以上のオペレーションフィールドを含む命令を解読し実行するデータ処理装置であって、
第１の前記オペレーションフィールドには実行すべき命令の流れを制御するオペコードが置かれ、
第２の前記オペレーションフィールドにはデータの転送を制御するオペコードが置かれ、
第３の前記オペレーションフィールドにはデータの算術論理演算を制御するオペコードが置かれ、
前記第１のオペレーションフィールドに置かれたオペコードを解読する第１の解読手段と、
前記第１の解読手段による解読結果に基づいて前記オペコードによって指定された実行すべき命令の流れの制御を実行する第１の実行手段と、
前記第２のオペレーションフィールドに置かれたオペコードを解読する第２の解読手段と、
前記第２の解読手段による解読結果に基づいて前記オペコードによって指定されたデータの転送の制御を実行する第２の実行手段と、
前記第３のオペレーションフィールドに置かれたオペコードを解読する第３の解読手段と、
前記第３の解読手段による解読結果に基づいて前記オペコードによって指定されたデータの算術論理演算を実行する第３の実行手段とを備えることを特徴とするＶＬＩＷプロセッサ。
【請求項２３】　前記第１のオペレーションフィールドの桁数は前記第２及び第３のオペレーションフィールドのいずれの桁数よりも小さいことを特徴とする請求項２２記載のＶＬＩＷプロセッサ。
【請求項２４】　前記第２のオペレーションフィールドの桁数は前記第３のオペレーションフィールドの桁数と等しいことを特徴とする請求項２３記載のＶＬＩＷプロセッサ。

【請求項２５】　前記第１、第２及び第３のオペレーションフィールドに置かれた各オペコードの桁数は等しいことを特徴とする請求項２４記載のＶＬＩＷプロセッサ。

【請求項２６】　３個以上のオペレーションフィールドとフォーマットフィールドを含む命令を解読し実行するデータ処理装置であって、

第１の前記オペレーションフィールドには実行すべき命令の流れを制御するオペコード又は定数が置かれ、

第２の前記オペレーションフィールドにはデータの転送を制御するオペコード又は定数が置かれ、

第３の前記オペレーションフィールドにはデータの算術論理演算を制御するオペコード又は定数が置かれ、

前記フォーマットフィールドには、前記第１、第２及び第３のオペレーションフィールドそれぞれに定数が置かれているか否かを指定するフォーマットコードが置かれ、

前記第１のオペレーションフィールドに置かれたオペコードを解読する第１の解読手段と、

前記第１の解読手段による解読結果に基づいて前記オペコードによって指定された実行すべき命令の流れの制御を実行する第１の実行手段と、

前記第２のオペレーションフィールドに置かれたオペコードを解読する第２の解読手段と、

前記第２の解読手段による解読結果に基づいて前記オペコードによって指定されたデータの転送の制御を実行する第２の実行手段と、

前記第３のオペレーションフィールドに置かれたオペコードを解読する第３の解読手段と、

前記第３の解読手段による解読結果に基づいて前記オペコードによって指定されたデータの算術論理演算を実行する第３の実行手段と、

前記フォーマットコードを解読するフォーマット解読手段と、

前記フォーマット解読手段により前記第１、第２及び第３の少なくとも１つのオペレーションフィールドに定数が置かれていると解読された場合に、その定数を取り出して記憶する定数記憶手段とを備えることを特徴とするＶＬＩＷプロセッサ。

【請求項２７】　前記命令が３２ビット長であることを特徴とする請求項２２又は２６記載のＶＬＩＷプロセッサ。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、ＶＬＩＷアーキテクチャを採るプロセッサに関し、特に、比較的短い語長であってコード効率の高い命令を実行するプロセッサに関する。

【０００２】

【従来の技術】近年のマルチメディア関連機器の需要の増大と電子機器の小型化に伴い、音声や画像データ等のマルチメディアデータを高速に処理できるマイクロプロセッサが必要とされている。この要求に応えるマイクロプロセッサとして、ＶＬＩＷ（Very Long Instruction Word）アーキテクチャを採るプロセッサ（以下、「ＶＬＩＷプロセッサ」という。）がある。

【０００３】ＶＬＩＷプロセッサは、内部に複数の演算ユニットを備え、１個のＶＬＩＷに置かれた複数のオペレーションを同時並列に実行する。このようなＶＬＩＷは、コンパイラよってソースプログラムにおけるオペレーションレベルでの並列性が検出されスケジューリングされた後に生成されたものである。ところが、特に機器組み込み用途においては、プログラムのコードサイズが問題となるために、２５６ビットの如く長いＶＬＩＷや、無動作命令（以下、「ＮＯＰ命令」という。）が頻繁に挿入されたコード効率の悪いＶＬＩＷは好ましくない。

【０００４】比較的短い語長の命令を実行する従来のＶＬＩＷプロセッサとして、最大２個のオペレーションを同時に指定することができる３２ビットの命令を実行するＶＬＩＷプロセッサがある（例えば、特開平９－２６８７８に開示されたデータ処理装置）。図１５（ａ）及び図１５（ｂ）は、上記従来技術における命令フォーマットを示し、それぞれ、同時に２個のオペレーションを指定する命令フォーマット、１個のオペレーションだけを指定する命令フォーマットを示す。この従来技術は、２ビットのフォーマットフィールド４１０の値によってその命令に置かれたオペレーションの数や実行順序を制御することで、コード効率を向上せんとするものである。

【０００５】

【発明が解決しようとする課題】しかしながら、上記従来技術では、３２ビット長の１個の命令で同時に指定できるオペレーションの数は最高で２個であり、その並列性は充分とは言えない。また、ある長さの語長を超える定数を用いた演算を行わせる場合には、命令のコード効率が低下してしまうという問題がある。例えば、３２ビットの定数をレジスタにセットするためにその定数を２つに分割し、定数の上位１６ビットをセットした後に下位１６ビットをセットした場合には、それらオペレーションの指定のためだけに２個の３２ビット長命令が消費されてしまう。

【０００６】そこで、本発明はかかる問題点に鑑みてなされたものであり、比較的短い語長の命令であって、かつ、同時に多くのオペレーションを指定することができるコード効率のよい構造を有する命令、例えば、３２ビット長の命令であれば３個以上のオペレーションを指定することができるような並列性の高い命令を実行するＶＬＩＷプロセッサを提供することを第１の目的とする。

【０００７】また、本発明の第２の目的は、比較的短い

語長の命令であって、かつ、比較的長い語長の定数を扱う場合においてもコード効率が低下しにくい構造を有する命令を実行するＶＬＩＷプロセッサを提供することである。

【０００８】

【課題を解決するための手段】上記第１の目的を達成するために本発明は、複数のオペレーションフィールドからなる命令を実行するＶＬＩＷプロセッサにおいて、前記オペレーションフィールドの大きさが不均一であり、かつ前記命令の命令語長は前記命令の持つオペレーションフィールドの数で割り切れないものであることを特徴とする。

【０００９】これによって、命令中の全てのオペレーションフィールドが同じ語長でなければならないという制限から解放され、コード効率のよい命令フォーマットを定義することが可能となるので、比較的短い語長の命令であって、かつ、同時に多くのオペレーションを指定することができるコード効率のよい構造を有する命令を実行するＶＬＩＷプロセッサが実現される。

【００１０】また、上記第２の目的を達成するために本発明は、２個以上のオペレーションフィールドを含む命令を解読し実行するＶＬＩＷプロセッサであって、第１の前記オペレーションフィールドにはオペレーションの種類を指定する１個のオペコードのみ又は定数のみが置かれ、第２の前記オペレーションフィールドには１個のオペコードとオペレーションの対象となるデータを指定する１個以上のオペランドとの組又は定数のみが置かれ、前記第１のオペレーションフィールドにオペコードが置かれた場合に前記オペコードを解読する第１の解読手段と、前記第１の解読手段による解読結果に基づいて前記オペコードによって指定されたオペレーションを実行する第１の実行手段と、前記第２のオペレーションフィールドにオペコードが置かれた場合に前記オペコードを解読する第２の解読手段と、前記第２の解読手段による解読結果に基づいて前記オペランドによって指定されたデータに対して前記オペコードによって指定されたオペレーションを実行する第２の実行手段とを備えることを特徴とする。

【００１１】これによって、命令中のあるオペレーションフィールドに無駄なコードを置く必要が生じた場合であっても、他のオペレーションで使用される定数で埋めておくことが可能となるので、比較的短い語長の命令であってもコード効率が低下しにくい構造を有する命令を実行するＶＬＩＷプロセッサが実現される。

【００１２】

【発明の実施の形態】以下、本発明に係るプロセッサの実施の形態について、図面を用いて詳細に説明する。なお、本明細書では、「命令」とは本プロセッサが同時並列に解読し実行するコード全体を意味し、「オペレーション」とは本プロセッサが並列に実行できる数値演算、論理演算、転送、分岐等の処理単位又はその処理単位を指定するためのコードを意味する。

（命令フォーマット）まず、本プロセッサが解読実行する命令の構造について説明する。

【００１３】本プロセッサは、ＶＬＩＷプロセッサであり、３２ビット固定長の命令を解読実行する。図１（ａ）は、本プロセッサが実行する命令５０のフィールド構成を示す図である。図１（ｂ）～図１（ｄ）は１６種類の命令フォーマットを示す図であり、そのうち、図１（ｂ）は３オペレーション、図１（ｃ）は２オペレーション、図１（ｄ）は１オペレーションを同時に指定できる命令フォーマットである。

【００１４】この命令５０は、３２ビット固定長であり、４ビットずつに区切られた８個のフィールド（上位よりＰ０．０フィールド５１、Ｐ１．０フィールド５２、…、Ｐ３．２フィールド５８）からなる。なお、Ｐ２．０フィールド５３～Ｐ２．２フィールド５５のグループをまとめて第１演算フィールド５９と呼び、Ｐ３．０フィールド５６～Ｐ３．２フィールド５８のグループをまとめて第２演算フィールド６０と呼ぶ。

【００１５】図１（ｂ）～図１（ｄ）において、“ｃｏｎｓｔ”は定数であり、これが用いられるオペレーションの種類によっては即値、絶対番地、ディスプレースメント等の数値定数や文字定数を意味する。“ｏｐ”はオペレーションの種類を指定するオペコードを、“Ｒｓ”はソースオペランドとなるレジスタを、“Ｒｄ”はデスティネーションオペランドとなるレジスタを、“ｃｃ”は本プロセッサが備える専用の３２ビットレジスタ（図３に示される定数レジスタ３６）の格納値を分岐先の絶対番地又は相対番地（ディスプレースメント）とする分岐オペレーションを指定するオペコードを意味する。

【００１６】また、これらコードの直後に添付された数値は、第１演算フィールド５９及び第２演算フィールド６０のいずれのオペレーションのために用いられるものであるかを示す。例えば、フォーマットコードが“６”である命令フォーマットの場合であれば、Ｐ１．０フィールド５２に置かれた４ビットの定数“ｃｏｎｓｔ１”とＰ２．１フィールド５４に置かれた４ビットの定数“ｃｏｎｓｔ１”とは結合され、８ビットの定数として第１演算フィールド５９のオペコード“ｏｐ１”に対応するソースオペランドになることを意味する。

【００１７】また、数値を伴わない定数“ｃｏｎｓｔ”は、本プロセッサが備える専用の３２ビットレジスタ（図３に示される定数レジスタ３６）に格納される定数を示す。例えば、フォーマットコードが“０”である命令フォーマットにおけるＰ１．０フィールド５２に置かれた４ビットの定数“ｃｏｎｓｔ”は、暗黙的に指定された定数レジスタ３６に格納される定数である。

【００１８】図２は、図１で用いられている３種類のオペコード“ｃｃ”、“ｏｐ１”及び“ｏｐ２”それぞれ

によって指定される具体的なオペレーションを説明する図である。４ビットのオペコード“ｃｃ”は、１６種類の分岐オペレーションの中の一つを指定する。１つの分岐オペレーションは、分岐条件と分岐形式によって特定される。分岐条件には、等しい（“ｅｑ”）、等しくない（“ｎｅｑ”）、より大きい（“ｇｔ”）等がある。分岐形式には、上記定数レジスタ３６の格納値を分岐先の絶対番地として分岐する形式（ニーモニック表示において“ｉ”が添付されていないもの）と相対番地として分岐する形式（ニーモニック表示において“ｉ”が添付されているもの）とがある。例えば、オペコード“ｅｑ”は、直前の比較結果が等しい場合に絶対番地指定による分岐を行なうオペレーションを意味し、オペコード“ｅｑｉ”は、直前の比較結果が等しい場合に相対番地指定による分岐を行なうオペレーションを意味する。

【００１９】４ビットのオペコード“ｏｐ１”は、“ａｄｄ”（加算）、“ｓｕｂ”（減算）、“ｍｕｌ”（乗算）、“ａｎｄ”（論理積）、“ｏｒ”（論理和）等の算術論理演算に属するオペレーションの一つを指定する場合と、“ｍｏｖ”（ワード（３２ビット）データの転送）、“ｍｏｖｈ”（ハーフワードデータの転送）、“ｍｏｖｂ”（バイトデータの転送）等のレジスタ・レジスタ間転送に属するオペレーションの一つを指定する場合とがある。

【００２０】４ビットのオペコード“ｏｐ２”は、上記オペコード“ｏｐ１”と同様の算術論理演算及びレジスタ・レジスタ間転送に加えて、“ｌｄ”（メモリからレジスタへの１ワードデータのロード）、“ｓｔ”（レジスタからメモリへのワードデータのストア）等のレジスタ・メモリ間転送に属するオペレーションの一つを指定する場合がある。

【００２１】次に、図１（ａ）に示された各フィールド５１、５２、５９、６０の特徴を説明する。Ｐ０．０フィールド５１は、この命令５０のフォーマットを特定する４ビットのフォーマットコードを置くためのフィールドであり、具体的には、図１（ｂ）～図１（ｄ）に示された１６種類の命令フォーマットの一つを特定する。

【００２２】Ｐ１．０フィールド５２は、定数又は分岐用のオペコードを置くためのフィールドである。このＰ１．０フィールド５２に定数が置かれた場合（フォーマットコード＝０、１、４～９の場合）には、その定数は、定数レジスタ３６に格納する対象となる場合（フォーマットコード＝０、１、４、５の場合）と、第１演算フィールド５９又は第２演算フィールド６０のオペランドの一部を構成する場合（フォーマットコード＝５、７、８、９、Ｂの場合）とがある。さらに、定数レジスタ３６に格納する対象となる場合には、その４ビットの定数のみが格納される場合（フォーマットコード＝０、１の場合）と、第１演算フィールド５９又は第２演算フィールド６０に置かれた１２ビットの定数と共に格納される場合（フォーマットコード＝４、５の場合）とがある。

【００２３】一方、このＰ１．０フィールド５２に分岐用のオペコード“ｃｃ”が置かれた場合（フォーマットコード＝２、３、Ａの場合）には、定数レジスタ３６に格納された定数を分岐先の絶対番地として、又は、相対番地（ディスプレースメント）として分岐することを意味する。第１演算フィールド５９は、本プロセッサと外部（メモリ）とのデータの転送を伴わないオペレーション（算術論理演算、レジスタ間転送）を指定するためのオペコードとオペランド（ソース及びデスティネーション）との組又は定数が置かれる。

【００２４】第２演算フィールド６０は、上記第１演算フィールド５９の場合に加えて、本プロセッサと外部（メモリ）とのデータの転送を伴うペレーション（レジスタ・メモリ間転送）を指定するためのオペコードとオペランドとの組が置かれることもある。なお、以上のようなオペレーションの種類の各フィールドへの割当ては、ノイマン型の本プロセッサにおいては２つ以上の分岐オペレーションを同時に実行する必要がないこと、本プロセッサと外部（メモリ）とのオペランドの入出力ポート（図３におけるオペランドアクセス部４０）を１つに限定していること等に基づく。

【００２５】ここで、図１（ｂ）～図１（ｄ）に示された命令フォーマットには以下の特徴がある。第１に、定数“ｃｏｎｓｔ”に着目して判るように、定数レジスタ３６に定数を格納させる命令フォーマットは次の３通りである。

（１）フォーマットコードが“０”又は“１”の場合：この命令では、Ｐ１．０フィールド５２に置かれた４ビットの定数が定数レジスタ３６に格納される。

（２）フォーマットコードが“４”の場合：この命令では、Ｐ１．０フィールド５２～Ｐ２．２フィールド５５に置かれた１６ビットの定数が定数レジスタ３６に格納される。

（３）フォーマットコードが“５”の場合：この命令では、Ｐ１．０フィールド５２とＰ３．０フィールド５６～Ｐ３．２フィールド５８に置かれた１６ビットの定数が定数レジスタ３６に格納される。

【００２６】第２に、本プロセッサでは、１個の命令に最大３つのオペレーションを指定することができるが、その場合には、図１（ｂ）に示された３オペレーション用の命令フォーマットから判るように、それら３つのオペレーションの種類は次のいずれかの組み合わせになる。

（１）４ビットの定数を定数レジスタ３６にセットするオペレーションと２個の汎用オペレーション（フォーマットコードが“０”、“１”の場合）

（２）定数レジスタ３６にセットされた値を絶対番地又は相対番地として分岐するオペレーションと２個の汎用

オペレーション（フォーマットコードが“2”、“3”の場合）
このように、本プロセッサの命令は、わずか３２ビット長でありながら最大3つのオペレーションを同時に指定することができるコード効率の高いフィールド構成を有している。
（プロセッサのハードウェア構成）次に、本プロセッサのハードウェア構成を説明する。
【００２７】図3は、本発明に係るプロセッサのハードウェア構成を示すブロック図である。本プロセッサは、上述したように、最大3つのオペレーションを並列実行するＶＬＩＷプロセッサであり、大きく分けて、命令レジスタ１０、解読部２０及び実行部３０から構成される。
【００２８】命令レジスタ１０は、命令フェッチ部３９から送られてきた１個の命令を保持する３２ビットのレジスタである。解読部２０は、命令レジスタ１０に保持された命令を解読し、その解読結果に応じた制御線を実行部３０に出力するものであり、大きく分けて、フォーマットデコーダ２１と命令デコーダ２２とからなる。
【００２９】命令デコーダ２２はさらに、Ｐ１．０フィールド１２に保持されたオペコード“ｃｃ”を解読しその結果に基づいてＰＣ部３３を制御する分岐デコーダ２３と、Ｐ２．０フィールド１３に保持されたオペコードを解読しその結果に基づいて第１演算部３７を制御する第１演算デコーダ２４と、Ｐ３．０フィールド１６に保持されたオペコードを解読しその結果に基づいて第２演算部３８及びオペランドアクセス部４０を制御する第２演算デコーダ２５とからなる。
【００３０】フォーマットデコーダ２１は、Ｐ０．０フィールド１１に保持された４ビットのフォーマットコードをデコードすることによって命令レジスタ１０に保持された命令のフォーマットが図１（ｂ）～図１（ｄ）に示された１６種類のうちのいずれであるかを特定し、その結果に応じて分岐デコーダ２３、第１演算デコーダ２４及び第２演算デコーダ２５による解読動作を許可又は禁止したり、実行部３０の定数レジスタ制御部３２を動作させたりする。
【００３１】なお、上記デコーダ２１、２３～２５は、基本的には１サイクルに１つのオペレーションを解読し、実行部３０に制御信号を与える。また、命令レジスタ１０と実行部３０を接続する２６ビットの定数信号線２６は、命令レジスタ１０に置かれた定数やオペランドを実行部３０に転送するためのバスである。実行部３０は、解読部２０での解読結果に基づいて、最大3つのオペレーションを並列実行する回路ユニットであり、実行制御部３１、ＰＣ部３３、レジスタ群３４、第１演算部３７、第２演算部３８、命令フェッチ部３９及びオペランドアクセス部４０からなる。なお、この実行部３０のうち定数レジスタ制御部３２、ＰＣ部３３及び定数レジスタ３６については、別の図面においてさらに詳細な構成を示している。
【００３２】実行制御部３１は、解読部２０での実行結果に基づいて実行部３０の各構成要素３３～４０を制御する制御回路や配線の総称であり、通常のプロセッサが備える構成要素（タイミング制御、動作許可禁止制御、ステータス管理、割り込み制御等の回路）の他に本プロセッサに特有の定数レジスタ制御部３２を有する。定数レジスタ制御部３２は、フォーマットデコーダ２１からの指示に基づいて命令レジスタ１０に保持された４ビット又は１６ビットの定数（ｃｏｎｓｔ）を定数レジスタ３６に格納する制御を行なう。
【００３３】ＰＣ（プログラムカウンタ）部３３は、分岐デコーダ２３による制御の下で、次に解読実行すべき命令が置かれている図示されていない外部メモリ上のアドレスを命令フェッチ部３９に出力する。命令フェッチ部３９は、３２ビットのＩＡ（インストラクションアドレス）バス及び３２ビットのＩＤ（インストラクションデータ）バスを通じて図示されていない外部メモリから命令ブロックをフェッチし、内部の命令キャッシュに保持すると共に、ＰＣ部３３から出力されたアドレスに相当する命令を命令レジスタ１０に供給する。
【００３４】レジスタ群３４は、１５個の３２ビット汎用レジスタ３５と１個の３２ビット定数レジスタ３６から構成される。これら１６個のレジスタ３５、３６に格納された値は、第１演算デコーダ２４及び第２演算デコーダ２５での解読結果に基づいて、第１演算部３７及び第２演算部３８に転送され、ここで演算が施され、又は、ここを単に通過した後に、レジスタ群３４又はオペランドアクセス部４０に送られる。なお、定数レジスタ３６に格納された値は、第１演算部３７及び第２演算部３８での演算に用いられる他に、ＰＣ部３３にも転送され、ここで分岐先となる有効アドレスを生成するために用いられる。
【００３５】第１演算部３７は、２個の３２ビットデータに対して算術論理演算を行なうＡＬＵと乗算を行う乗算器とを内部に有し、第１演算デコーダ２４による制御の下で２種類のオペレーション（算術論理演算とレジスタ間転送）を実行する。第２演算部３８も、第１演算部３７と同様に、２個の３２ビットデータに対して算術論理演算を行なうＡＬＵと乗算を行う乗算器とを内部に有し、第２演算デコーダ２５による制御の下で２種類のオペレーション（算術論理演算とレジスタ間転送）を実行する。
【００３６】オペランドアクセス部４０は、第２演算デコーダによる制御の下でレジスタ群３４と図示されていない外部メモリとの間でオペランドの転送を行なう回路であり、そのオペランドやオペランドアドレスを保持するバッファを内部に有する。具体的には、例えば、命令レジスタ１０のＰ３．１フィールド１６にオペコード

“１ｄ”が置かれていた場合には、外部メモリに置かれていた１ワードのデータがオペランドアクセス部４０を経てレジスタ群３４のいずれかのレジスタにロードされ、また、オペコード“ｓｔ”が置かれていた場合には、レジスタ群３４のいずれかのレジスタの格納値が外部メモリにストアされる。

【００３７】上記ＰＣ部３３、レジスタ群３４、第１演算部３７、第２演算部３８及びオペランドアクセス部４０は、図示されるように、内部バス（Ｌ１バス、Ｒ１バス、Ｌ２バス、Ｒ２バス、Ｄ１バス、Ｄ２バス）で接続されている。なお、Ｌ１バス及びＲ１バスはそれぞれ第１演算部３７の２つの入力ポートに、Ｌ２バス及びＲ２バスはそれぞれ第２演算部３８の２つの入力ポートに、Ｄ１バス及びＤ２バスはそれぞれ第１演算部３７及び第２演算部３８の出力ポートに接続されている。

（定数レジスタ３６及びその周辺回路の詳細な構成）次に、定数レジスタ３６及びその周辺回路について詳細に説明する。

【００３８】図４は、定数レジスタ３６及びその周辺回路の詳細な構成と接続関係を示すブロック図である。なお、図中の固定値（“０”）２７は、定数“０”を示す４本の信号線の固定的な配線を意味する。定数レジスタ制御部３２は、５個の３入力セレクタ３２ａ～３２ｅと３個の４入力セレクタ３２ｆ～３２ｈとからなり、定数レジスタ３６は、８個の４ビット幅レジスタ３６ａ～３６ｈからなる。なお、各入出力データは並列４ビットである。

【００３９】定数レジスタ制御部３２は、フォーマットデコーダ２１及び命令デコーダ２２からの制御信号に従って上記８個の入力セレクタ３２ａ～３２ｈを制御することで、以下に示す４通りの格納方法のいずれかの方法により、命令レジスタ１０に保持された定数又はゼロを定数レジスタ３６に格納させる。図５（ａ）～図５（ｄ）は、その４通りの格納方法を説明する図である。

【００４０】図５（ａ）は、フォーマットデコーダ２１によってＰ０．０フィールド１１に保持された値が“０”又は“１”であると解読された場合の格納方法を示す。これは、Ｐ１．０フィールド１２に置かれた４ビットの定数のみを定数レジスタ３６に格納する場合に相当する。具体的には、定数レジスタ３６に保持されたデータを４ビット単位で上位にシフトさせると同時に、命令レジスタ１０のＰ１．０フィールド１２に保持された４ビットの定数を定数レジスタ３６の最下位の４ビットレジスタ３６ｈに格納する。

【００４１】図５（ｂ）は、フォーマットデコーダ２１によってＰ０．０フィールド１１に保持された値が“４”であると解読された場合の格納方法を示す。これは、Ｐ１．０フィールド１２～Ｐ２．２フィールド１５に置かれた１６ビットの定数を定数レジスタ３６に格納する場合に相当する。具体的には、定数レジスタ３６の下位１６ビット３６ｅ～３６ｈに保持されたデータを上位１６ビット３６ａ～３６ｄにシフトさせると同時に、命令レジスタ１０のＰ１．０フィールド１２～Ｐ２．２フィールド１５に保持された１６ビットの定数を定数レジスタ３６の下位１６ビット３６ｅ～３６ｈに格納する。

【００４２】図５（ｃ）は、フォーマットデコーダ２１によってＰ０．０フィールド１１に保持された値が“５”であると解読された場合の格納方法を示す。これは、Ｐ１．０フィールド１２とＰ３．０フィールド１６～Ｐ３．２フィールド１８に置かれた１６ビットの定数を定数レジスタ３６に格納する場合に相当する。具体的には、定数レジスタ３６の下位１６ビット３６ｅ～３６ｈに保持されたデータを上位１６ビット３６ａ～３６ｄにシフトさせると同時に、命令レジスタ１０のＰ１．０フィールド１２とＰ３．０フィールド１６～Ｐ３．２フィールド１８に保持された１６ビットの定数を定数レジスタ３６の下位１６ビット３６ｅ～３６ｈに格納する。

【００４３】図５（ｄ）は、フォーマットデコーダ２１によってＰ０．０フィールド１１に保持された値が“２”、“３”及び“Ａ”のいずれかであると解読された場合又は命令デコーダ２２によってＰ２．１フィールド１４、Ｐ２．２フィールド１５、Ｐ３．２フィールド１７及びＰ３．３フィールド１８の少なくとも一つに定数レジスタ（Ｒ１５）が指定されている場合の格納方法を示す。これは、Ｐ１．０フィールド１２に置かれた分岐オペレーション、第１演算フィールド５９及び第２演算フィールド６０の少なくとも一つのオペレーションによって定数レジスタ３６の格納値が使用された（読み出された）後に、定数レジスタ３６にオールゼロを格納する（定数レジスタ３６をクリアする）場合に相当する。

【００４４】具体的には、定数レジスタ３６の格納値がＰＣ部３３、第１演算部３７及び第２演算部３８のいずれかに読み出された直後に、３２ビットの定数“０”を定数レジスタ３６に格納する。なお、定数レジスタ３６の使用後にクリアしておくのは、定数レジスタ３６には常にゼロ拡張された値が格納されていることを保証するためである。ここで、ゼロ拡張とは、ある数値の有効桁数が一定の桁数に満たない場合に、その有効桁より上位の桁全てをゼロで埋める処理をいう。

【００４５】以上のように、命令レジスタ１０のＰ０．０フィールド１１の値が“０”、“１”、“４”、“５”の場合には、定数レジスタ３６に既に格納された定数をシフトさせながら新たな定数が定数レジスタ３６に格納される。また、定数レジスタ３６は、その格納値が一旦読み出されて使用されると、その内容は消去される。このようにして、定数レジスタ３６は、その内容が読み出されるまで、次々に格納される定数を蓄積していくことができる。

（ＰＣ部３３の詳細な構成）次に、ＰＣ部３３の詳細な

構成を説明する。

【００４６】図６は、ＰＣ部３３の詳細な構成を示すブロック図である。ＰＣ部３３は、定数“４”を示す固定的な配線である固定値（“４”）３３ａ、２入力セレクタ３３ｂ、加算器３３ｃ、次に解読実行すべき命令のアドレスを保持するＰＣ３３ｄ及び４入力セレクタ３３ｅから構成される。このＰＣ部３３では、解読部２０からの制御信号に従ってセレクタ３３ｂ、３３ｅが動作することにより、以下の３通りの値のいずれかが有効アドレスとしてセレクタ３３ｅから命令フェッチ部３９に出力される。

（１）ＰＣ３３ｄの内容に“４”を加算した値

これは、分岐しないで順次に実行する場合、即ち、解読実行された命令に分岐オペレーションが指定されていない場合に相当する。なお、“４”を加算するのは、１つの命令の長さが４バイト（３２ビット）であることによる。

（２）ＰＣ３３ｄの内容に定数レジスタ３６の内容を加算した値

これは、定数レジスタ３６の内容を相対番地として分岐する場合、例えば、Ｐ１．０フィールド１２によって相対番地による分岐が指定されていると分岐デコーダ２３が解読した場合が該当する。

（３）定数レジスタ３６の内容

これは、定数レジスタ３６の内容を絶対番地として分岐する場合、例えば、Ｐ１．０フィールド１２によって絶対番地による分岐が指定されていると分岐デコーダ２３が解読した場合が該当する。

【００４７】以上のように、このＰＣ部３３は、専用の加算器３３ｃを備え、定数レジスタ３６に保持された値を直接用いる構成となっているので、第１演算部３７や第２演算部３８での演算とは独立並行して、定数レジスタ３６の格納値を絶対番地又は相対番地として分岐する実行制御を行なうことができる。

（プロセッサの動作）次に、具体的な命令を解読実行した場合の本プロセッサの動作について説明する。

【００４８】図７は、３２ビットの定数を扱う処理の一例を示すフローチャートである。本図には、レジスタＲ０とＲ１との格納値の差を求め（ステップＳ８０）、その結果にレジスタＲ２の格納値を掛け（ステップＳ８１）、さらにその結果に３２ビットの定数“０ｘ８７６５４３２１”（１６進数の“８７６５４３２１”）を加え（ステップＳ８２、Ｓ８３）、最後にレジスタＲ３をクリアしておく（ステップＳ８５）という処理が示されている。

【００４９】図８は、図７に示された処理内容を本プロセッサに行なわせるプログラムの例を示す図である。このプログラムは、３個の命令７１～７３から構成されている。１行が１個の命令に相当し、各命令の内容は各フィールドに置かれたニーモニックで表現されている。なお、定数は全て１６進数で表現されている。また、“ｆｍｔｎ（ｎ＝０～Ｆ）”はフォーマットコード“ｎ”を示し、“Ｒｎ（ｎ＝０～１５）”はレジスタ群３４の中の１つのレジスタを示す。なお、“Ｒ１５”は定数レジスタ３６を意味する。

【００５０】図９は、図８に示されたプログラムを実行した場合の本プロセッサの動作を示すタイミングチャートである。本図には、クロックサイクル、汎用レジスタＲ０～Ｒ３及び定数レジスタＲ１５の内容、４つのバスＬ１、Ｒ１，Ｌ２，Ｒ２を流れるデータが示されている。上記図８及び図９を用いて、各命令７１～７３ごとの本プロセッサの動作を説明する。

（命令７１）命令７１が命令レジスタ１０にロードされると、本プロセッサは図９のクロックサイクルｔ０～ｔ１に示された動作をする。

【００５１】フォーマットデコーダ２１は、命令レジスタ１０のＰ０．０フィールド１１の値（“ｆｍｔ４”）から、この命令はフォーマットコードが“４”の２オペレーション命令であると判断し、以下の２つのオペレーションが並列実行されるように実行部３０を制御する。

（１）第１のオペレーション

定数レジスタ制御部３２は、内部の８個のセレクタ３２ａ～３２ｈを制御することで、図５（ｂ）に示された格納方法により、Ｐ１．０フィールド１２～Ｐ２．２フィールド１５に保持された１６ビットの定数（０ｘ８７６５）を定数レジスタ３６の下位１６ビットに格納する。その結果、図９のクロックサイクルｔ０～ｔ１に示されるように、定数レジスタＲ１５の内容は、それまでの“０ｘ００００００００”から“０ｘ０００８７６５”に変化する。

（２）第２のオペレーション

第２演算部３８は、汎用レジスタＲ０の内容（“０ｘ３３３３３３３”）と汎用レジスタＲ１の内容（“０ｘ２２２２２２２２”）とを入力とし、ここで減算した後に、その結果を再び汎用レジスタＲ０に格納する。その結果、図９のクロックサイクルｔ０～ｔ１に示されるように、汎用レジスタＲ０の内容は、それまでの“０ｘ３３３３３３３”から“０ｘ１１１１１１１”に変化する。

（命令７２）次に、命令７２が命令レジスタ１０にロードされると、本プロセッサは図９のクロックサイクルｔ１～ｔ２に示された動作をする。

【００５２】フォーマットデコーダ２１は、上記命令７１の場合と同様に、命令レジスタ１０のＰ０．０フィールド１１の値（“ｆｍｔ４”）から、この命令はフォーマットコードが“４”の２オペレーション命令であると判断し、以下の２つのオペレーションが並列実行されるように実行部３０を制御する。

（１）第１のオペレーション

定数レジスタ制御部３２は、内部の８個のセレクタ３２

ａ～３２ｈを制御することで、図５（ｂ）に示された格納方法により、Ｐ１．０フィールド１２～Ｐ２．２フィールド１５に保持された１６ビットの定数（０ｘ４３２１）を定数レジスタ３６の下位１６ビットに格納する。その結果、図９のクロックサイクルｔ１～ｔ２に示されるように、定数レジスタＲ１５の内容は、それまでの“０ｘ００００８７６５”から“０ｘ８７６５４３２１”に変化する。

（２）第２のオペレーション

第２演算部３８は、汎用レジスタＲ２の内容（“０ｘ００００００００４”）と汎用レジスタＲ０の内容（“０ｘ１１１１１１１１”）とを入力とし、ここで乗算した後に、その結果を再び汎用レジスタＲ０に格納する。その結果、図９のクロックサイクルｔ１～ｔ２に示されるように、汎用レジスタＲ０の内容は、それまでの“０ｘ１１１１１１１１”から“０ｘ４４４４４４４４”に変化する。

（命令７３）最後に、命令７３が命令レジスタ１０にロードされると、本プロセッサは図９のクロックサイクルｔ２～ｔ３に示された動作をする。

【００５３】フォーマットデコーダ２１は、命令レジスタ１０のＰ０．０フィールド１１の値（“ｆｍｔ７”）から、この命令はフォーマットコードが“７”の２オペレーション命令であると判断し、以下の２つのオペレーションが並列実行されるように実行部３０を制御する。

（１）第１のオペレーション

第１演算部３７は、定数レジスタＲ１５の内容（“０ｘ８７６５４３２１”））値と汎用レジスタＲ０の内容（“０ｘ４４４４４４４４）とを入力とし、それらを加算した後に、その結果を再び汎用レジスタＲ０に格納する。その結果、図９のクロックサイクルｔ２～ｔ３に示されるように、汎用レジスタＲ０の内容は、それまでの“０ｘ４４４４４４４４”から“０ｘＣＢＡ９８７６５”に変化し、定数レジスタＲ１５の内容はクリアされる。

（２）第２のオペレーション

第２演算部３８は、Ｐ１．０フィールド１２とＰ３．１フィールド１７に分割して置かれた８ビットの定数（“０ｘ００”）を入力とし、そのまま通過させて、汎用レジスタＲ３に格納する。その結果、図９のクロックサイクルｔ２～ｔ３に示されるように、汎用レジスタＲ３の内容は、それまでの“０ｘＦＥＤＣＢＡ９８”から“０ｘ００００００００”に変化する。

【００５４】以上のようにして、本プロセッサにおいて、３２ビットの定数“０ｘ８７６５４３２１”は、２個の命令７１、７２に跨って分割配置され、順次定数レジスタ３６にシフトされながら格納された後に、第３番目の命令７３によって利用された。このようにして、図７のフローチャートに示された処理が３個の命令７１～７３によって実行される。次に、１６ビットの定数を扱う別のプログラムを用いて本プロセッサの動作を説明する。

【００５５】図１０は、１６ビットの定数を扱うプログラムの例を示す図である。このプログラムは、５個の命令７４～７８から構成されている。各命令７１～７３ごとの本プロセッサの動作は以下の通りである。

（命令７４）命令７４が命令レジスタ１０にロードされると、フォーマットデコーダ２１は、命令レジスタ１０のＰ０．０フィールド１１の値（“ｆｍｔ０”）から、この命令はフォーマットコードが“０”の３オペレーション命令であると判断し、以下の３つのオペレーションが並列実行されるように実行部３０を制御する。

（１）第１のオペレーション

定数レジスタ制御部３２は、内部の８個のセレクタ３２ａ～３２ｈを制御することで、図５（ａ）に示された格納方法により、Ｐ１．０フィールド１２に保持された４ビットの定数（“０ｘ８”）を定数レジスタ３６の最下位の４ビットレジスタ３６ｈに格納する。

（２）第２のオペレーション

第１演算部３７は、汎用レジスタＲ６の値を入力とし、そのまま通過させて、汎用レジスタＲ１に格納する。

（３）第３のオペレーション

同様に、第２演算部３８は、汎用レジスタＲ７の値を入力とし、そのまま通過させて、汎用レジスタＲ２に格納する。

（命令７５）同様にして、命令７５が命令レジスタ１０にロードされると、フォーマットデコーダ２１は、この命令はフォーマットコードが“０”の３オペレーション命令であると判断し、以下の３つのオペレーションが並列実行されるように実行部３０を制御する。

（１）第１のオペレーション

定数レジスタ制御部３２は、内部の８個のセレクタ３２ａ～３２ｈを制御することで、図５（ａ）に示された格納方法により、Ｐ１．０フィールド１２に保持された４ビットの定数（“０ｘ７”）を定数レジスタ３６の最下位４ビットレジスタ３６ｈに格納する。この結果、定数レジスタ３６の下位８ビットには定数“０ｘ８７”がセットされる。

（２）第２のオペレーション

第１演算部３７は、汎用レジスタＲ０とＲ１の値を入力とし、ここで加算した後に、その結果を再び汎用レジスタＲ１に格納する。

（３）第３のオペレーション

同様に、第２演算部３８は、汎用レジスタＲ０とＲ２の値を入力とし、ここで加算した後に、その結果を再び汎用レジスタＲ２に格納する。

（命令７６、命令７７）同様にして、命令７６、７７が実行されることにより、定数レジスタ３６の下位１６ビットには定数“０ｘ８７６５”がセットされる。

（命令７８）命令７８が命令レジスタ１０にロードされ

ると、本プロセッサは、図８に示された命令７３の場合と同様の動作をする。

【００５６】以上のようにして、本プロセッサにおいては、１６ビットの定数“０ｘ８７６５”は、４個の命令７４～７７に跨って分割配置され、順次定数レジスタ３６にシフトされながら格納された後に、第５番目の命令７８によって利用された。

（通常のプロセッサとの比較）次に、上記図８及び図１０に示されたプログラムと同一内容の処理を通常のプロセッサに行なわせた場合について説明し、本発明に係るプロセッサと比較する。なお、ここでいう通常のプロセッサとは、本発明に係るプロセッサの定数レジスタ３６や定数レジスタ制御部３２の如く、分割された定数を蓄積して格納する手段のみを有しないプロセッサをいい、３２ビット固定長の命令を実行するものとする。

【００５７】図１１（ａ）は、この通常のプロセッサが実行する命令のフィールド定義を示し、図１１（ｂ）は、その命令のフォーマットを示す。つまり、通常のプロセッサは、３種類の２オペレーション命令１０１～１０３と１種類の１オペレーション命令１０４を実行するものとする。図１２は、図８に示されたプログラムと同一内容の処理、即ち、図７のフローチャートに示された処理を通常のプロセッサに行なわせるプログラムの例である。

【００５８】図１２と図８とを比較して判るように、通常のプロセッサ用のプログラムは、本発明に係るプロセッサ用のものよりも２個の命令だけ多くなっている。なお、命令１０５、１０６にｎｏｐコードが含まれるのは、命令１０６は命令１０５での演算結果を用いるので、これらの命令を並列に実行させることができないからである。また、１個の定数“０ｘ８７６５４３２１”を上位１６ビットと下位１６ビットの２つに分割して定数レジスタＲｉにセットしているのは（命令１０７、１０８）、３２ビットの１個の命令の中に、セット命令のオペコードと３２ビットの定数の両方を同時に配置することは不可能だからである。

【００５９】同様に、図１３は、図１０に示されたプログラムと同一内容の処理を通常のプロセッサに行なわせるプログラムの例である。図１３と図１０とを比較して判るように、通常のプロセッサ用のプログラムは、本発明に係るプロセッサ用のものよりも１個の命令だけ多くなっている。以上のように、本発明に係るプロセッサが実行する命令は、３２ビットという比較的短い語長でありながら最大３つのオペレーションを同時に指定することができるコード効率の高いフィールド構成を有している。

【００６０】そして、本発明に係るプロセッサによれば、１６ビットや３２ビットの定数が複数の命令に跨って分割配置されていても、それらは定数レジスタ３６に蓄積して格納されることで元の定数に復元され、分岐や算術演算等のオペレーションに使用される。つまり、命令中に生じた小さな領域であっても、定数を分割して埋めておくことができるので、通常のプロセッサに実行させる場合よりもプログラムのコードサイズは縮小される。以上、本発明に係るプロセッサについて、実施形態に基づいて説明したが、本発明はこれら実施形態に限られないことは勿論である。即ち、

（１）上記実施の形態では、命令５０は、３２ビット長であり、８個の４ビット長のフィールドからなり、最大３個のオペレーションを指定することができる構造を有したが、本発明は、これら数値に限定されるものではない。

【００６１】例えば、上記フィールド構成にさらに、１個の４ビット長のオペコードと１個の４ビット長のオペランドとの組からなる８ビット長のフィールドを加えた合計４０ビット長の命令とすることもできる。これによって、４０ビットという比較的短い語長の命令でありながら最大４つのオペレーションを同時に実行させることができるコード効率の高い命令が定義される。

（２）また、３２ビット固定長命令によって３個のオペレーションを同時に指定することができる命令の構造として、図１（ａ）に示された命令構造の他に、図１４（ａ）～図１４（ｄ）に示された命令構造とすることもできる。これら図中において、縦線の最小間隔は１ビット長を示し、“ｆｍｔ”はフォーマットフィールドを示す。

【００６２】図１４（ａ）に示された構造の命令であれば、上記実施形態の命令に比較し、より多くの命令フォーマットを定義することができる点、及び、３つのオペレーションフィールドそれぞれに少なくとも１個のオペランドを置くことができる点において優る。図１４（ｂ）～図１４（ｄ）に示された構造の命令であれば、上記実施形態の命令に比較し、２個のオペコード（“ｏｐ２”、“ｏｐ３”）の桁数が大きいので、より多くの種類のオペレーションを定義することができる点において優る。

（３）また、上記実施の形態の命令５０では、暗黙的なオペランド（定数レジスタ３６の格納値）を用いるフィールドは１箇所だけであったが、これに限定されるものではなく、２箇所以上であってもよい。新たな命令フォーマットを定義することで対応すればよい。

（４）また、上記実施の形態では、数値定数を扱う例が示されたが、文字定数であってもよいことは言うまでもない。複数の命令に跨って分割配置された文字定数であっても、定数レジスタ３６への複数回の格納によって、桁数の長い元の文字定数が復元されるからである。

（５）また、上記実施の形態では、図１（ｂ）～図１（ｄ）の命令フォーマットから判るように、１個の命令によって定数レジスタ３６に格納させることができる定数の桁数は４ビット及び１６ビットのいずれかであった

が、本発明はこの桁数に限定されるものではない。例えば、１２ビットや２８ビットの定数を定数レジスタ３６に格納するための命令フォーマットを定義してもよい。そのためには、定数レジスタ３６の周辺回路の接続関係を変更すればよい。

【００６３】

【発明の効果】以上の説明から明らかなように、本発明は、複数のオペレーションフィールドからなる命令を実行するＶＬＩＷプロセッサにおいて、前記オペレーションフィールドの大きさが不均一であり、かつ前記命令の命令語長は前記命令の持つオペレーションフィールドの数で割り切れないものであることを特徴とする。

【００６４】これによって、命令中の全てのオペレーションフィールドが同じ語長でなければならないという制限から解放され、コード効率のよい命令フォーマットを定義することが可能となるので、比較的短い語長の命令であって、かつ、同時に多くのオペレーションを指定することができるコード効率のよい構造を有する命令を実行するＶＬＩＷプロセッサが実現される。

【００６５】ここで、複数のオペレーションフィールドからなる命令を実行するＶＬＩＷプロセッサにおいて、前記オペレーションフィールドの大きさが不均一であり、かつ前記命令は３つのオペレーションフィールドを３２ビットの命令語長中に持つものであるとすることもできる。これによって、３２ビット長で３個のオペレーションを指定することができる並列性の高い命令を実行するＶＬＩＷプロセッサが実現される。

【００６６】また、複数のオペレーションフィールドからなる命令を実行するＶＬＩＷプロセッサにおいて、前記オペレーションフィールドのうち少なくとも１つはオペランドの数が異なるものであるとすることもできる。これによって、命令中の全てのオペレーションフィールドが同じ個数のオペランドを有さなければならないという制限から解放されるので、コード効率のよい命令フォーマットを定義することが可能となる。

【００６７】また、複数のオペレーションフィールドからなる命令を実行するＶＬＩＷプロセッサにおいて、前記オペレーションフィールドのうち１つはオペコードのみからなるとすることもできる。これによって、命令中の全てのオペレーションフィールドにオペコードとオペランドとの組が置かれる場合に比べ、命令の語長は短縮されるので、コード効率のよい構造を有する命令を実行するＶＬＩＷプロセッサが実現される。

【００６８】また、２個以上のオペレーションフィールドを含む命令を解読し実行するＶＬＩＷプロセッサであって、第１の前記オペレーションフィールドにはオペレーションの種類を指定する１個のオペコードのみが置かれ、第２の前記オペレーションフィールドには１個のオペコードとオペレーションの対象となるデータを指定する１個以上のオペランドとの組が置かれ、前記第１のオペレーションフィールドに置かれたオペコードを解読する第１の解読手段と、前記第１の解読手段による解読結果に基づいて前記オペコードによって指定されたオペレーションを実行する第１の実行手段と、前記第２のオペレーションフィールドに置かれたオペコードを解読する第２の解読手段と、前記第２の解読手段による解読結果に基づいて前記オペランドによって指定されたデータに対して前記オペコードによって指定されたオペレーションを実行する第２の実行手段とを備えるとすることもできる。

【００６９】これによって、命令中の少なくとも１つのオペレーションには明示的なオペランドを伴わないオペコードのみを置くことができるので、命令語長は短縮される。また、前記第１のオペレーションフィールドに置かれたオペコードの桁数は前記第２のオペレーションフィールドに置かれたオペコードの桁数と等しいとすることもできる。

【００７０】これによって、命令中に置かれる全てのオペコードの桁数を共通にすることができるので、デコーダ回路等が簡単化される。また、前記命令に含まれるオペレーションフィールドは３個であり、第３の前記オペレーションフィールドは前記第２のオペレーションフィールドと同じ桁数であって１個のオペコードと１個以上のオペランドとの組が置かれ、前記ＶＬＩＷプロセッサはさらに、前記第３のオペレーションフィールドにオペコードが置かれた場合に前記オペコードを解読する第３の解読手段と、前記第３の解読手段による解読結果に基づいて前記オペランドによって指定されたデータに対して前記オペコードによって指定されたオペレーションを実行する第３の実行手段とを備えるとすることもできる。

【００７１】これによって、同時に３個のオペレーションを実行するＶＬＩＷプロセッサが実現される。また、前記第１の実行手段は、実行すべき命令の流れを制御するとすることもできる。これによって、一般的に多くの桁数を必要としない分岐オペレーションが桁数の小さいオペレーションフィールドに割り当てられるので、コード効率のよい命令セットが定義される。

【００７２】また、前記第２の実行手段は、前記第２のオペランドフィールドに置かれたオペランドによって指定されたデータの転送を制御し、前記第３の実行手段は、前記第３のオペランドフィールドに置かれたオペランドによって指定されたデータの算術論理演算を実行するとすることもできる。これによって、外部メモリとのデータ転送は命令中の１個のオペレーションだけによって指定されることになるので、ＶＬＩＷプロセッサが備えるべきオペランドアクセス回路は簡単化される。

【００７３】また、２個以上のオペレーションフィールドを含む命令を解読し実行するＶＬＩＷプロセッサであって、第１の前記オペレーションフィールドにはオペレ

ーションの種類を指定する１個のオペコードのみ又は定数のみが置かれ、第２の前記オペレーションフィールドには１個のオペコードとオペレーションの対象となるデータを指定する１個以上のオペランドとの組又は定数のみが置かれ、前記第１のオペレーションフィールドにオペコードが置かれた場合に前記オペコードを解読する第１の解読手段と、前記第１の解読手段による解読結果に基づいて前記オペコードによって指定されたオペレーションを実行する第１の実行手段と、前記第２のオペレーションフィールドにオペコードが置かれた場合に前記オペコードを解読する第２の解読手段と、前記第２の解読手段による解読結果に基づいて前記オペランドによって指定されたデータに対して前記オペコードによって指定されたオペレーションを実行する第２の実行手段とを備えるとすることもできる。

【００７４】これによって、命令中のあるオペレーションフィールドに無駄なコードを置く必要が生じた場合であっても、他のオペレーションで使用される定数で埋めておくことが可能となるので、比較的短い語長の命令であってもコード効率が低下しにくい構造を有する命令を実行するＶＬＩＷプロセッサが実現される。また、前記命令はさらに、前記第１及び第２のオペレーションフィールドそれぞれに定数のみが置かれているか否かを指定するフォーマットコードが置かれたフォーマットフィールドを含み、前記ＶＬＩＷプロセッサはさらに、前記フォーマットコードを解読するフォーマット解読手段と、前記フォーマット解読手段により前記第１、第２及び第３の少なくとも１つのオペレーションフィールドに定数のみが置かれていると解読された場合に、その定数を取り出して記憶する定数記憶手段とを備えるとすることもできる。

【００７５】これによって、オペレーションフィールドに埋められた定数は定数記憶手段に格納され、他の命令中に置かれたオペレーションによってその定数を利用することが可能となるので、比較的短い語長の命令であって、かつ、比較的長い語長の定数を扱う場合においてもコード効率の低下が回避される。また、前記フォーマットフィールドの桁数、前記第１のオペレーションフィールドの桁数、前記第２及び第３のオペレーションフィールドに置かれたオペコードの桁数、前記第２及び第３のオペレーションフィールドに置かれた各オペランドの桁数は、いずれもｎビットであるとすることもできる。

【００７６】これによって、１個の命令を構成する全てのフィールドの桁数が同じになるので、ＶＬＩＷプロセッサの内部回路が簡単化される。以上のように、本発明によって、比較的短い語長の命令であって、かつ、同時に多くのオペレーションを指定することができるコード効率のよい構造を有する命令を実行するＶＬＩＷプロセッサが実現され、特にマルチメディアデータを処理する組み込み用途のプロセッサとしてその実用的価値は大きい。

【図面の簡単な説明】

【図１】図１（ａ）は、本発明に係るプロセッサが実行する命令のフィールド構成を示す図である。図１（ｂ）～図１（ｄ）は、１６種類の命令フォーマットを示す図である。図１（ｂ）は３オペレーション、図１（ｃ）は２オペレーション、図１（ｄ）は１オペレーションを同時に指定できる命令フォーマットである。

【図２】図１で用いられている３種類のオペコード“ｃｃ”、“ｏｐ１”及び“ｏｐ２”それぞれによって指定される具体的なオペレーションを説明する図である。

【図３】同プロセッサのハードウェア構成を示すブロック図である。

【図４】同プロセッサの定数レジスタ３６及びその周辺回路の詳細な構成を示すブロック図である。

【図５】図４に示された定数レジスタ制御部３２による定数の格納方法を示す図である。図５（ａ）はフォーマットコードが“０”又は“１”である場合、図５（ｂ）はフォーマットデコードが“４”である場合、図５（ｃ）はフォーマットデコードが“５”である場合、図５（ｄ）はフォーマットコードが“２”、“３”及び“Ａ”のいずれかである場合又は定数レジスタ３６の格納値がオペランドとして指定されている場合の格納方法を示す。

【図６】同プロセッサのＰＣ部３３の詳細な構成を示すブロック図である。

【図７】３２ビットの定数を扱う処理の一例を示すフローチャートである。

【図８】図７に示された処理を同プロセッサに行なわせるプログラムの例を示す図である。

【図９】図９は、図８に示されたプログラムを実行した場合の本プロセッサの動作を示すタイミングチャートである。

【図１０】１６ビットの定数を扱う処理を同プロセッサに行なわせるプログラムの例を示す図である。

【図１１】図１１（ａ）は、通常のプロセッサが実行する命令のフィールド定義を示す図である。図１１（ｂ）は、同命令フォーマットを示す図である。

【図１２】図８に示されたプログラムと同一内容の処理を上記通常のプロセッサに行なわせるプログラムの例を示す図である。

【図１３】図１０に示されたプログラムと同一内容の処理を上記通常のプロセッサに行なわせるプログラムの例を示す図である。

【図１４】図１４（ａ）～図１４（ｄ）は、本発明のＶＬＩＷプロセッサにかかる命令構造の変形例を示す図である。

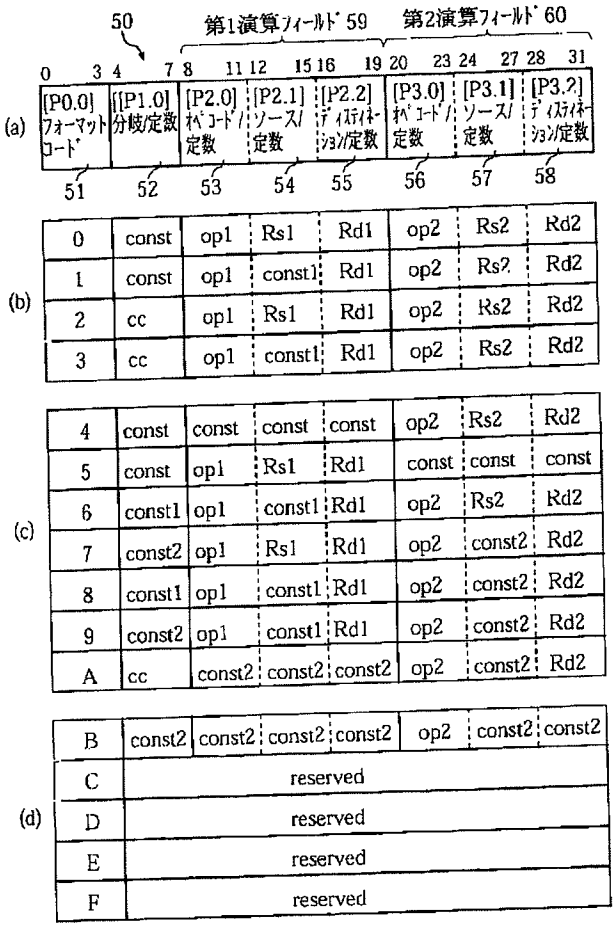
【図１５】図１５（ａ）及び図１５（ｂ）は、従来技術における命令フォーマットを示し、それぞれ、同時に２個のオペレーションを指定する命令フォーマット、１個

のオペレーションだけを指定する命令フォーマットを示す図である。

【符号の説明】

10　命令レジスタ
20　解読部
21　フォーマットデコーダ
22　命令デコーダ
23　分岐デコーダ
24　第1演算デコーダ
25　第2演算デコーダ
30　実行部
31　実行制御部
32　定数レジスタ制御部
32a～32h　セレクタ
33　PC部
33a　固定値“4”
33b、33e　セレクタ
33c　加算器
33d　PC
34　レジスタ群
35　汎用レジスタR0～R14
36　定数レジスタR15
36a～36h　4ビット幅レジスタ
37　第1演算部
38　第2演算部
39　命令フェッチ部
40　オペランドアクセス部
41　セレクタ
50　命令
51～58　命令フィールド
59　第1演算フィールド
60　第2演算フィルド

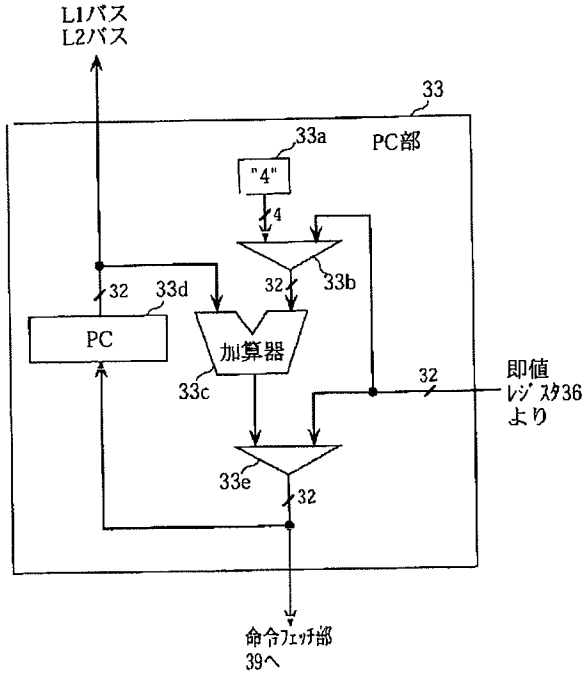
【図1】



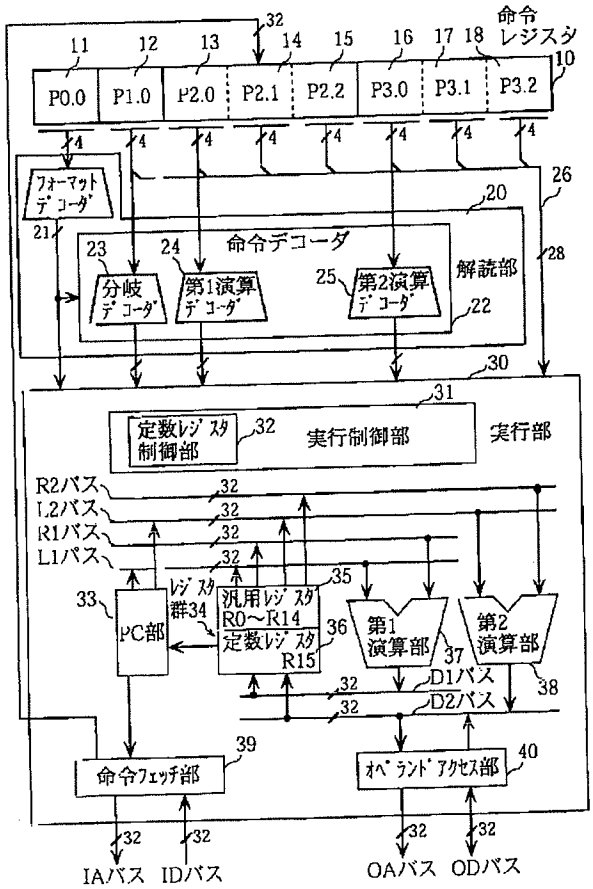
【図2】

記号	オペレーション	ニーモニック表示
cc	分岐	eq, eqi, ne, nei, gt, gti, ・・・
op1	算術論理演算	add, sub, mul, and, or, ・・・
	レジスタ間転送	mov, movh, movb
op2	算術論理演算	add, sub, mul, and, or, ・・・
	レジスタ間転送	mov, movh, movb
	レジスタ・メモリ間転送	ld, ldh, ldb, st, sth, stb

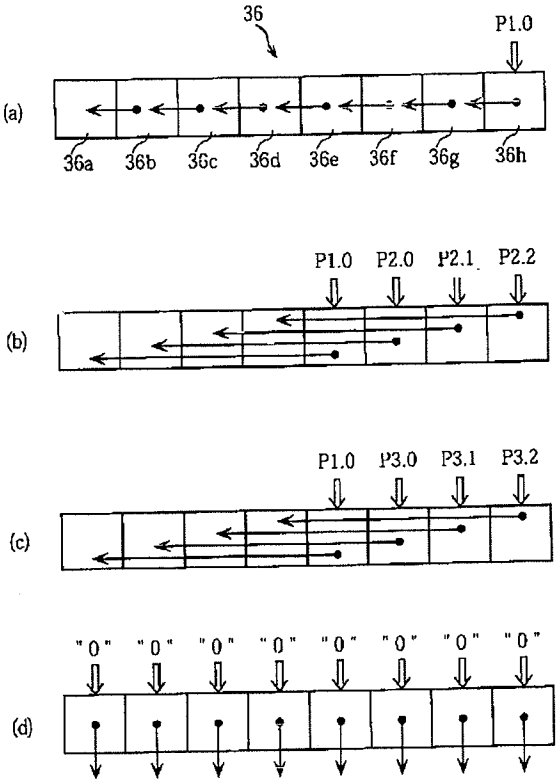
【図6】



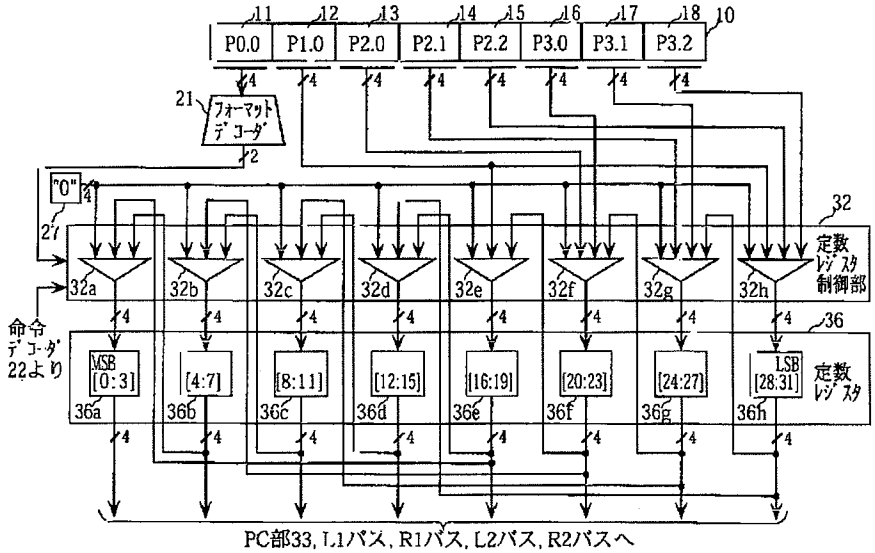
【図3】


命令レジスタ
P0.0
P1.0
P2.0
P2.1
P2.2
P3.0
P3.1
P3.2
フォーマットデコーダ
命令デコーダ
分岐デコーダ
第1演算デコーダ
第2演算デコーダ
解読部
定数レジスタ制御部
実行制御部
実行部
R2バス
L2バス
R1バス
L1バス
PC部
レジスタ群34
汎用レジスタR0～R14
定数レジスタR15
第1演算部
第2演算部
D1バス
D2バス
命令フェッチ部
オペランドアクセス部
IAバス
IDバス
OAバス
ODバス

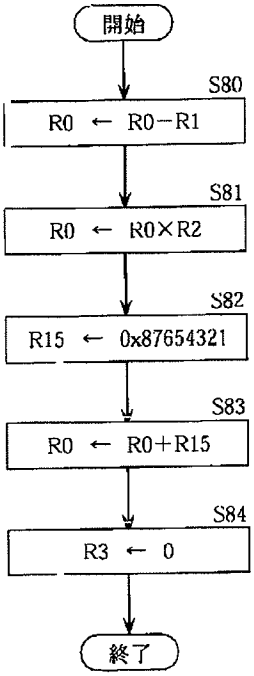
【図5】


(a)
(b)
(c)
(d)
P1.0
P2.0
P2.1
P2.2
P3.0
P3.1
P3.2
"0"

【図4】


P0.0
P1.0
P2.0
P2.1
P2.2
P3.0
P3.1
P3.2
フォーマットデコーダ
"0"
定数レジスタ制御部
命令デコーダ22より
MSB[0:3]
[4:7]
[8:11]
[12:15]
[16:19]
[20:23]
[24:27]
LSB[28:31]
定数レジスタ
PC部33, L1バス, R1バス, L2バス, R2バスへ

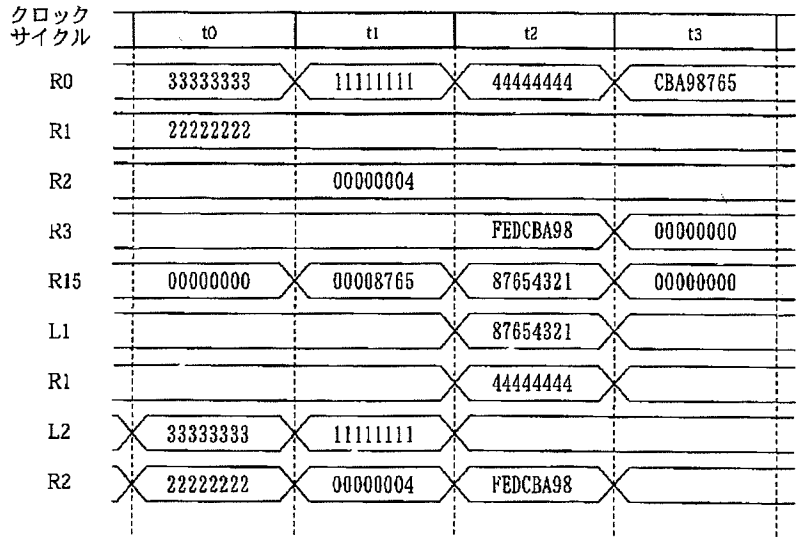
【図7】


開始
S80
R0 ← R0−R1
S81
R0 ← R0×R2
S82
R15 ← 0x87654321
S83
R0 ← R0+R15
S84
R3 ← 0
終了

【図8】

P0.0	P1.0	P2.0	P2.1	P2.2	P3.0	P3.1	P3.2	
fmt 4	0x8765				Sub	R1	R0	71
fmt 4	0x4321				mul	R2	R0	72
fmt 7	0x0	add	R15	R0	mov	0x0	R3	73

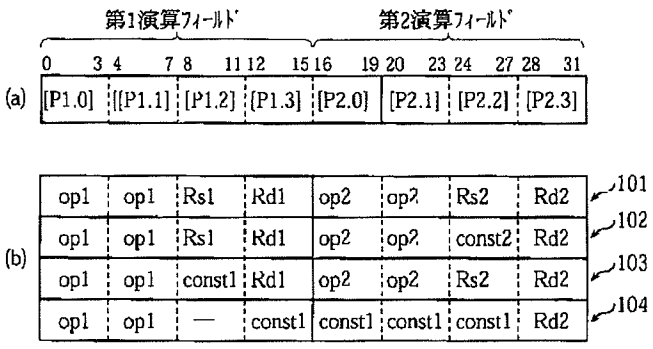
【図9】



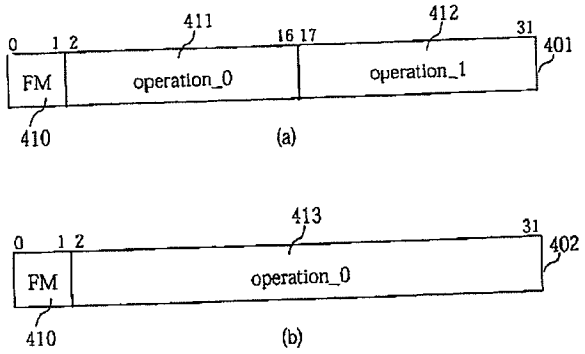
【図10】

P0.0	P1.0	P2.0	P2.1	P2.2	P3.0	P3.1	P3.2	
fmt 0	0x8	mov	R6	R1	mov	R7	R2	74
fmt 0	0x7	add	R0	R1	add	R0	R2	75
fmt 0	0x6	mul	R6	R1	sub	R7	R2	76
fmt 0	0x5	mov	R8	R4	mov	R9	R5	77
fmt 7	0x0	add	R15	R0	mov	0x0	R3	78

【図11】



【図15】



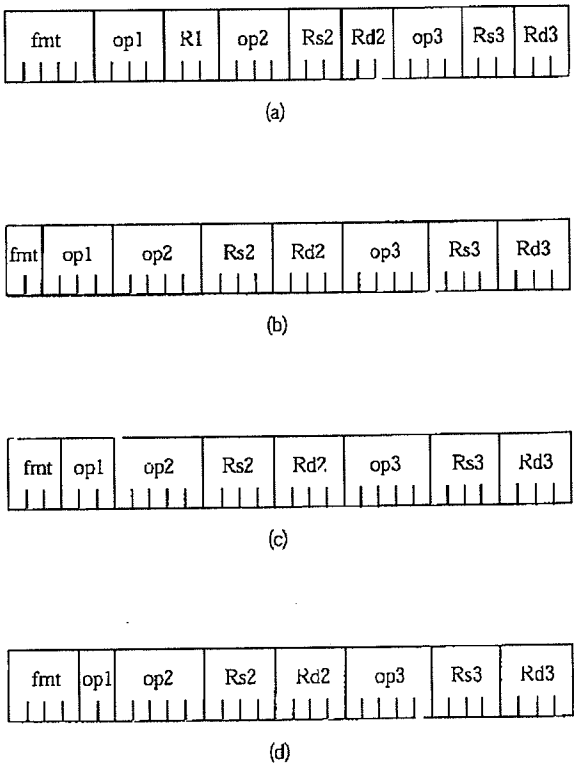
【図12】

P1.0	P1.1	P1.2	P1.3	P2.0	P2.1	P2.2	P2.3	
nop		—	—	sub		R1	R0	105
nop		—	—	mul		R2	R0	106
set hi		—	0x8765				R15	107
set lo		—	0x4321				R15	108
add		R15	R0	mov		0x0	R3	109

【図13】

P1.0	P1.1	P1.2	P1.3	P2.0	P2.1	P2.2	P2.3	
mov		R6	R1	mov		R7	R2	110
add		R0	R1	add		R0	R2	111
mul		R6	R1	sub		R7	R2	112
mov		R8	R4	mov		R9	R5	113
add		—	0x8765				R0	114
nop		—	—	mov		0x0	R3	115

【図１４】


fmt
op1
R1
op2
Rs2
Rd2
op3
Rs3
Rd3
(a)
fmt
op1
op2
Rs2
Rd2
op3
Rs3
Rd3
(b)
fmt
op1
op2
Rs2
Rd2
op3
Rs3
Rd3
(c)
fmt
op1
op2
Rs2
Rd2
op3
Rs3
Rd3
(d)